# Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ StudentID: \_\_\_\_\_\_\_\_\_\_\_

## Activity 1: Styling Text

How do we style the following text based properties in an element with CSS:

* text color
* color
* type of font
* font-family
* size of text
* font-size
* **boldness** and *slopey-ness*
* **font-style**
* underline, ~~strike-through~~ and drop-shadows
* text-decoration
* spacing between letters and lines
* letter-spacing
* left, right, centered and justified alignment
* text-align

When using an exotic font (ie one not found on all computers), what options do we have today?

List out different types of font in the order according to your preference (most to least favorite) , if one is not available, it will be replaced by the one right after it

## 

## 

## Activity 2: Styling Background and Borders

How do we set the background color?

Background-color

What happens if you specify a background image and a background color together?

The image change to that color

How many different ways can we set a background image in an element?

4 ways: solid color, linear gradients, radial gradients, repeating color gradients

What's the difference between a linear gradient and a radial gradient?

Linear: change from one color to other in a straight direction

Radial: change from one color to the other from center of the screen to outside

How do we set the border thickness, style and color?

Border: ‘thickness’px style color

How do we make corners rounded? What about drop-shadows?

Border – radius

Text-shadow, box-shadow

## 

## 

## Activity 3: CSS Box Model

What is the box model? What do the properties width, padding, border and margin refer to?

Box model create a container for text/ image (content)

Width: horizontal size of content

Padding: space from border to content

Border: properties to border lines

Margin: space from border to other elements outside of box

Boxes have 4 sides (going clockwise: top, right, bottom, left), but many dimension related styles have a **compact** version where only 1, 2 or 3 values are specified. Explain how the browser interprets each variant.

It will automatically adjust the last variable to keep the box shape a rectangle

How wide will this element's visible box be: <div id='fixed-content'></div> (CSS shown below):

|  |
| --- |
| #fixed-content  {  width: 100px;  border: grey solid 5px;  padding: 20px;  margin: 20px;  }  <div id='fixed-content'></div> |

Width: 150px

What difference does the new CSS3 box-sizing:border-box setting make to the box sizing model?

Reduce the width to fit inside the parent element

How wide will this element's visible box be: <div id='adaptable-content'></div> (CSS shown below), In particular, how large will it be displayed on a large screen (eg on a 27" iMac) or a small screen (eg an older "palette" monitor?):

|  |
| --- |
| #adaptable-content  {  width: 50%;  min-width: 600px;  max-width: 1800px;  margin: 0 auto;  }  <div id='adaptable-content'></div> |

Width will be 50% of the screen

If screen smaller than 600px, keep width at 600px

If screen larger than 1800px, keep width at 1800px

## 

How does the CSS3 calc() and box-sizing styles help us fine-tune box dimensions?

|  |
| --- |
| #fine-tuned-content  {  box-sizing: border-box;  width: calc(100% - 60px);  }  <div id='fine-tuned-content'></div> |

## 

## Box sizing: border-box – shrink the width to fit parent element

## Calc help create 30px padding

## Activity 4: Floating Page Elements, Inline and Flexible Blocks



Block elements usually take up the entire width of the parent element, but there are times we want to float an image to the left or right (eg these mugs) and have text float around it (eg these paragraphs).



When we set an element to float:left or float:right any block elements that come after it are placed in the gaps. This might be what you want, but more often than not it isn't. Paragraphs will break up and page footers will look like inline elements (see below).

|  |
| --- |
| Footer content | T&C | Links | Map | Uh oh, look at this image → |

To "stop the floats", you will need an element that has the style clear:both to restore the normal flow layout.

<http://www.w3schools.com/cssref/pr_class_float.asp>

<http://www.w3schools.com/cssref/pr_class_clear.asp>

CSS3 allows us to have the best of both worlds with floating blocks. Elements with display:inline-block will have inline flow properties but retain a box shape.

|  |
| --- |
| #keep-content-together  {  display:inline-block;  }  <div id='keep-content-together'></div> |

Internally, elements will not be reshaped and don't get squashed like the above paragraphs do.

<http://www.w3schools.com/cssref/pr_class_display.asp>

A new advanced family of styles (supported in modern browsers only) is flex and grid which allows you to create flexible elements that resize according to rules laid down by their parent element and their sibling elements. These styles are likely to dominate page layout design in the near future.

[https://www.w3schools.com/cssref/css3\_pr\_flex.asp](http://www.w3schools.com/cssref/css3_pr_flex.asp)

<https://www.w3schools.com/cssref/pr_grid.asp>

## Activity 5: Positioning

The default style for positioning is static. What does this mean?

Position don’t change over time

What other positioning styles are there? What happens to the original space that the element occupied in each case?

Relative – similar to static but the position depends on its original place

Absolute – relative to the parent tag

Fixed – relative to browser window, won’t scroll with the page

Sticky – scroll with the page

How do top, bottom, left and right help us to adjust the position of a non-static element?

Adjust position relative to the window

## 

## Activity 6: Display and Visibility

To make elements "disappear" we have at least two options:

|  |
| --- |
| #gollum, #bilbo, #frodo, #harry, .friends-of-harry  {  visibility: hidden;  /\* to make visible, use visibility: visible; \*/  }  #tardis  {  display: none;  /\* to re-display, use display: block; OR inline-block; OR inline; etc \*/  } |

What is the difference between the two styles above?

Display: none – delete the space that contained the element

Visibility: hidden – still keep the space of the element